

3101-451 DP Miles S  
Stockbridge

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2003-6179

(P2003-6179A)

(43) 公開日 平成15年1月10日 (2003.1.10)

(51) IntCl. <sup>7</sup>	識別記号	F I	テーマコード(参考)
G 0 6 F 15/78	5 1 0	G 0 6 F 15/78	5 1 0 P 5 B 0 1 1
1/04	3 0 1	1/04	3 0 1 C 5 B 0 6 2
1/32		1/00	3 3 2 Z 5 B 0 7 9

031431 U.S. PTO  
10/772438

審査請求 未請求 請求項の数12 O L (全 10 頁)

(21) 出願番号 特願2001-184224(P2001-184224)

(22) 出願日 平成13年6月19日 (2001.6.19)

(71) 出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72) 発明者 石橋 孝一郎

東京都国分寺市東恋ヶ窪一丁目280番地

株式会社日立製作所中央研究所内

(72) 発明者 入江 直彦

東京都国分寺市東恋ヶ窪一丁目280番地

株式会社日立製作所中央研究所内

(74) 代理人 100075096

弁理士 作田 康夫

最終頁に続く

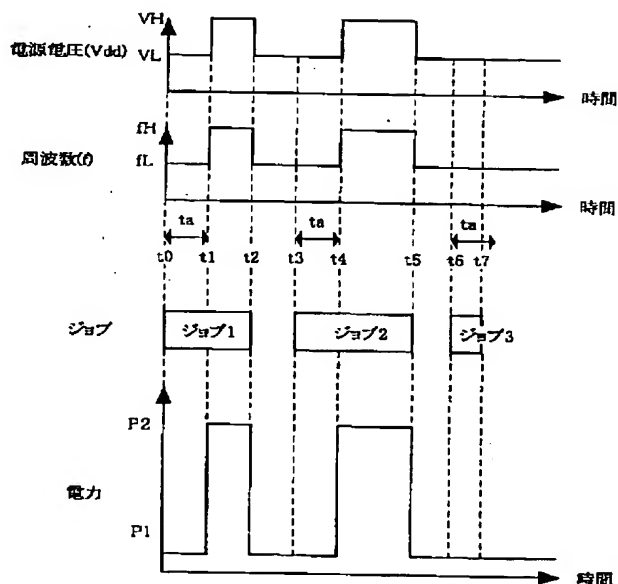
(54) 【発明の名称】 半導体装置および半導体装置の動作モード制御方法

## (57) 【要約】

【課題】 OSに依存することなく、汎用性の高い周波数制御方式を実現し、マイクロプロセッサの低電力化と高性能化を両立させる。

【解決手段】 ジョブが実行される場合には、低い周波数で動作を開始し、あらかじめ決められた時間経過した後、そのジョブの実行が継続していた場合には、自動的に周波数を高くしてそのジョブを実行する手段を備える。

図1



BEST AVAILABLE COPY

【特許請求の範囲】

【請求項1】複数の命令が連続的に実行されるジョブを実行するプロセッサと、

上記プロセッサの動作モードを制御する制御部とを有し、

上記プロセッサが上記ジョブの実行を開始するにあたって、上記制御部は上記プロセッサの動作モードを第1モードに設定し、

上記プロセッサが上記ジョブの実行を開始した後、所定の時間経過後に、上記制御部は上記プロセッサの動作モードを第2モードに設定し、

上記第2モードにおいて上記プロセッサは、上記第1モードにおいて可能な動作速度よりも高速動作可能である半導体装置。

【請求項2】請求項1において、

上記第1モードにおいて、上記プロセッサは第1の動作周波数で上記ジョブを実行し、

上記第2モードにおいて、上記プロセッサは上記第1の動作周波数よりも高い第2の動作周波数で上記ジョブを実行する半導体装置。

【請求項3】請求項2において、

上記第1モードにおいて、上記プロセッサは第1の電源電圧で上記ジョブを実行し、

上記第2モードにおいて、上記プロセッサは上記第1の電源電圧よりも高い第2の電源電圧で上記ジョブを実行する半導体装置。

【請求項4】請求項2において、

上記第1モードにおいて、上記プロセッサを構成するトランジスタの基板またはウェルに第1の基板電位を印加し、

上記第2モードにおいて、上記トランジスタの基板またはウェルに上記第1の基板電位の絶対値よりもその絶対値の小さい第2の基板電位を印加する半導体装置。

【請求項5】請求項1において、

上記制御部は、上記プロセッサの動作モードを制御する制御回路と、上記所定の時間についての情報を記憶する第1のレジスタとを備えた半導体装置。

【請求項6】請求項5において、

電源投入時またはシステムリセット時に、上記所定の時間についての情報が外部の記憶装置から上記第1のレジスタに転送される半導体装置。

【請求項7】請求項5において、

上記プロセッサは、ポインタがアイドルアドレスを指すことによって上記ジョブの実行を終了し、

上記制御部は上記アイドルアドレスを記憶する第2のレジスタと、上記第2のレジスタに記憶されたアイドルアドレスと上記ポインタとを比較する比較器とを有し、電源投入時またはシステムリセット時に、上記アイドルアドレスが外部の記憶装置から上記第2のレジスタに転送され、

上記制御回路は、上記比較器が上記第2のレジスタに記憶されたアイドルアドレスと上記ポインタとが不一致であることを示す出力を受けて、上記プロセッサが上記ジョブの実行を開始したことを判定する半導体装置。

05 【請求項8】請求項5において、

上記プロセッサにクロック信号を供給するクロック信号供給回路を備え、

上記プロセッサは上記プロセッサへの上記クロック信号の供給を制御する制御信号を出力し、

10 上記制御回路は、上記プロセッサが上記プロセッサへの上記クロック信号の供給を開始させる制御信号を出力したのに応じて、上記プロセッサが上記ジョブの実行を開始したことを判定する半導体装置。

【請求項9】請求項1において、

15 上記プロセッサは第1のアプリケーションの処理に対応する第1のジョブと、第2のアプリケーションの処理に対応する第2のジョブとを実行し、

上記プロセッサが上記第1のジョブを実行する場合には、上記第1のジョブの実行を開始した後、第1の時間

20 経過後に上記制御部は上記プロセッサの動作モードを上記第1モードから上記第2モードに変更し、

上記プロセッサが上記第2のジョブを実行する場合には、上記第2のジョブの実行を開始した後、上記第1の

25 時間と異なる第2の時間経過後に上記制御部は上記プロセッサの動作モードを上記第1モードから上記第2モードに変更する半導体装置。

【請求項10】複数の命令が連続的に実行されるジョブを実行するプロセッサを含む半導体装置の動作モード制御方法であって、

30 電源投入もしくはシステムリセット時に上記半導体装置のレジスタに時間情報を格納し、

第1の動作モードで上記ジョブの実行を開始し、

上記ジョブの実行の開始後、上記時間情報により指示された時間経過後に上記第1の動作モードから上記第2の

35 モードに変更する半導体装置の動作モード制御方法。

【請求項11】請求項10において、

上記時間情報は、外部の不揮発メモリまたは上記半導体装置の内蔵する不揮発メモリから上記半導体装置に転送される半導体装置の動作モード制御方法。

40 【請求項12】請求項10において、

上記プロセッサは第1のアプリケーションの処理に対応する第1のジョブと、第2のアプリケーションの処理に対応する第2のジョブとを実行し、

45 上記電源投入もしくは上記システムリセット時に上記半導体装置のレジスタに上記第1のジョブについての第1の時間情報を格納し、

上記電源投入もしくは上記システムリセット時に上記半導体装置のレジスタに上記第2のジョブについての第1の時間情報を格納し、

50 上記プロセッサが上記第1のジョブを実行時には、上記

第1のジョブの実行の開始後、上記第1の時間情報により指示された時間経過後に上記第1の動作モードから上記第2の動作モードに変更し、  
上記プロセッサが上記第2のジョブを実行時には、上記第2のジョブの実行の開始後、上記第2の時間情報により指示された時間経過後に上記第1の動作モードから上記第2の動作モードに変更する半導体装置の動作モード制御方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、マイクロプロセッサとその制御方法に関する。

【0002】

【従来の技術】近年、マイクロプロセッサの高速化は著しく、約1年で2倍の高速化が達成されている。一方で、高速化に伴う消費電力の増大の問題が顕在化してきた。特に、マイクロプロセッサを携帯機器システムに用いる場合において、消費電力の増大は電池寿命を短くする。携帯機器システムは電池寿命を延ばすことを最優先させるため、プロセッサが実行できる最大の周波数よりも周波数を落として動作させることによって、その消費電力を削減させる。しかしながら、この方式ではプロセッサは高度な情報処理が実行できない。

【0003】このような問題に対し、プロセッサが実行する負荷が小さいときには電源電圧と周波数とを下げてその消費電力を低減し、負荷が大きいときには電源電圧と周波数とを上げてその性能向上を図る方式がすでに提案されている。

【0004】このような方式の従来例としては2000アイイーイーイー、インターナショナル、ソリッド、ステート、サーキット、コンファランス、ダイジェスト、オブ、テクニカル、ペーパーズ(2000, IEEE International Solid-State Circuits, Digest of Technical Papers)292頁から293頁に示されている。この従来例では、プロセッサで実行するアプリケーションの要求する締め切りに合わせて周波数と電源電圧を制御する。必要な周波数を割り出すためにオペレーティングシステム（以後OSという）を用い、その必要な周波数を実行できる電源電圧を発生させるための回路と電源等からなるシステムを提案している。

【0005】

【発明が解決しようとする課題】上記従来例では、最適な周波数を割り出すためにOSを用いているため、以下の問題が生じる。

(1) OSはこの制御を行うために最適な周波数を割り出すための計算が必要となる。OS自身はマイクロプロセッサが走らせるため、マイクロプロセッサに制御のためのオーバーヘッドが生じ、その消費電力が増大する。

(2) この制御方式では、マイクロプロセッサとOSの両方の協調動作が必要であるため、両者が同じ電源周波

数制御システムに対応していないと動作できない。しかしながら、実際には世の中にはすでにさまざまなOSとさまざまなマイクロプロセッサが存在しており、実際にさまざまな機器に使用されている。現実にはこれらのOSとマイクロプロセッサにこの制御方式を広く適用するには、規格化等の問題が生じることが予想される。

【0006】そこで、本発明はOSに依存することなく、汎用性の高い周波数制御方式を実現し、マイクロプロセッサの低電力化と高性能化を両立させる。

【0007】

【課題を解決するための手段】このため、本発明は、以下の手段を用いる。

【0008】複数の命令が連続的に実行されるジョブを実行するプロセッサと、プロセッサの動作モードを制御する制御部とを有する半導体装置であって、プロセッサがジョブの実行を開始するにあたって、制御部はプロセッサの動作モードを第1モードに設定し、プロセッサがジョブの実行を開始した後、所定の時間経過後に、制御部は上記プロセッサの動作モードを第2モードに設定する。ここで、第2モードにおいてプロセッサは、第1モードにおいて可能な動作速度よりも高速動作可能とする。具体的には、第1モードよりも第2モードの動作周波数を高くする、第1モードよりも第2モードの動作周波数を高くするとともに電源電圧を高くする、第1モードよりも第2モードの動作周波数を高くするとともに基板バイアスの絶対値を小さくする。

【0009】また、制御部は、プロセッサのポインタのアドレスや、クロック供給回路に出力する制御信号の状態によりプロセッサの状態を検知して、プロセッサ上で動作するOSとは無関係に動作モードを制御可能にする。

【0010】さらに、第1モードから第2モードに変更すべき所定の時間についての情報を外部のROMや内蔵の不揮発性メモリに格納し、電源投入時またはシステムリセット時に制御部のレジスタに転送することにより、上記所定の時間をプロセッサが処理すべきアプリケーションに適した時間に設定することができる。

【0011】

【発明の実施の形態】以下、本発明を実施例に基づき説明する。

【0012】図1は本発明の第1の実施例である電源電圧及び周波数の双方を制御する例を示す波形図である。この例では、プロセッサはジョブ1、ジョブ3、ジョブ3を実行する。ここでジョブとはプロセッサで実行される実質的に連続した命令列である。

【0013】ジョブ1は時刻 $t_0$ に実行を開始される。このとき、プロセッサは低い動作周波数 $f_L$ と低い電源電圧 $V_L$ で実行を開始する。その後、ある一定の時間 $t_a$ が経過した時刻 $t_1$ でプロセッサは時間 $t_a$ の経過を検知して電源電圧と周波数を変化させる。時刻 $t_1$ 以降で

は、プロセッサは高い動作周波数 $f_H$ と高い電源電圧 $V_H$ でジョブ1を実行する。最後に時刻 $t_2$ でジョブ1の実行が終了する。同様に、ジョブ2は時刻 $t_3$ で周波数 $f_L$ と電源電圧 $V_L$ でスタートし、時間 $t_a$ が経過した時刻 $t_4$ から、プロセッサは周波数 $f_H$ と電源電圧 $V_H$ でジョブ2を実行する。ジョブ2は時刻 $t_5$ で終了する。次に、ジョブ3が時刻 $t_6$ でスタートする。ジョブ3は周波数 $f_L$ と電源電圧 $V_L$ でスタートするが比較的短いジョブであり、時間 $t_a$ を経過しないで終了してしまう。すなわち、ジョブ3は周波数と電源電圧を周波数 $f_H$ と電源電圧 $V_H$ に変更する制御がなされることなく、時刻 $t_6$ で終了する。

【0014】本例における消費電力が図1に示されている。CMOS回路の電力を $P$ とすると、その値は周波数と電圧の2乗の積に比例する、すなわち $P \propto f * V^2$ で表されることがよく知られている。

【0015】一例として、 $V_H = 2V_L$ 、 $f_H = 2f_L$ なる関係が成り立つとする。

【0016】この場合、動作周波数 $f_L$ 、電源電圧 $V_L$ で動作する場合の消費電力 $P_1$ は、 $P_1 \propto f_L * V_L^2$ となり、動作周波数 $f_H$ 、電源電圧 $V_H$ で動作する場合の消費電力 $P_2$ は、 $P_2 \propto f_H * V_H^2 = 2f_L * (2V_L)^2 = 8P_1$ となる。このように、 $P_1$ は $P_2$ の $1/8$ の電力である。電源電圧を下げずに単に周波数のみを下げる方式では電力は周波数に比例した分しか下がらない（この例では $1/2$ ）ので、電源と周波数を同時に制御する方式は効果的である。この電源電圧と周波数とを制御する制御方式は、消費電力 $P_1$ の時間が長いほど平均的な消費電力を下げるができる。ここで、電力 $P_1$ のモードを低電力モード、電力 $P_2$ のモードを高速モードと定義する。

【0017】本制御方式では、あらかじめ決められた時間 $t_a$ を設定して、低電力動作と高速動作とを切り分ける。ジョブを実行し始めたときには低電力モードになっている。比較的負荷の軽いジョブは低電力モード下で短時間に実行を完了してしまうので、小さい消費電力で実行が完了する。一方、負荷の重いジョブの場合は低電力モードの期間（時間 $t_a$ ）ではジョブが終了しない。ここで、一定の時間 $t_a$ に達すると自動的に高速モードに移行する。これにより、負荷が重い場合には高速の周波数で動作させることができるので、プロセッサの最高性能を利用することができる。また、ジョブを連続してとぎれなく実行させる場合も考えられるが、このような場合はそもそもプロセッサの能力を最大に使う場合であって、高速モードが維持されたままでよい。

【0018】時間 $t_a$ の情報は半導体集積回路内のレジスタ等に蓄えておけばよい。集積回路内のハードウェアは時間 $t_a$ の経過を監視して、低電力モードから高速モードに自動的に移行させることができる。したがって、簡便な方法により、マイクロプロセッサの低電力化と高性能化を両立させることができる。図2にかかるハードウェアの構成例を示す。

【0019】半導体集積回路20は、ソフトウェアを実行するためのプロセッサ（CPU）25と、プロセッサの動作モードを制御するための制御部26、クロック供給回路（クロック供給回路をPLLで代表させ、以下、PLLと表記する）23、電源回路24を有している。さらに、制御部26は、時間 $t_a$ の情報を保持する $t_a$ レジスタ21と、PLL23及び電源回路24を制御する制御回路22とを有している。制御回路22はジョブの実行状態を監視し、ジョブが開始された場合にはPLL23と電源回路24を制御してプロセッサの動作周波数 $f$ を $f_L$ 、電源電圧 $V_{dd}$ を $V_L$ にする。さらに、制御回路22は、 $t_a$ レジスタ21に保持された時間 $t_a$ とジョブの実行時間とを比較し、ジョブの実行時間が $t_a$ に達するとPLL23と電源回路24を制御して周波数と電源電圧をそれぞれ $f_H$ と $V_H$ にする。また、ジョブが終了したら周波数と電源電圧を $f_L$ と $V_L$ にもどす。

【0020】なお、電源回路24は必ずしも半導体集積回路内になくてもよく、出力電圧を制御できることの可能な電源素子を用いてもよい。図3にハードウェアの別の構成例を示す。この例では、システムのハードウェア構成をより詳細に示している。1は半導体集積回路装置、2は電源回路、3はブートROMである。

【0021】システムの制御方式は図2に示した例と同様である。この例では、電源回路2が半導体集積回路装置1とは別の回路に存在することが特徴である。半導体集積回路装置1の制御回路4は電源回路2に制御信号を送り、電源回路2が発生する電源電圧 $V_{dd}$ を制御する。また、この例では、ブートROM3と半導体集積回路装置1が接続されており、ブートROM3はシステムを立ち上げたときに必要な情報を半導体集積回路装置1に取り込む動きをしている。ブートROM3に時間 $t_a$ の情報を記憶しておき、システムの電源投入時またはシステムリセット時にレジスタ5に入力すればよい。なお、時間 $t_a$ の情報としては、時間情報でもよく、クロック数でもよい。前者の場合はタイマー6で計測し、後者の場合は図示しないカウンタで計測すればよい。このように構成することにより、その後OSを半導体集積回路装置1のメモリ等にインストールする場合でもOSに依存しないで、本発明の動作を実行することができる。

【0022】ここで、ブートROMは一般的にはEEPROM等の不揮発メモリが使われる場合が多いが、不揮発な記憶媒体であればなんでもよく、フラッシュメモリ、電池バックアップのRAM、フロッピーディスク、ハードディスク等でも、システム立ち上げ時に自動的に内容が読み込まれるようになっていればよい。また、図3の例ではブートROMは半導体集積回路装置1の外部にあるが、半導体集積回路装置1上に形成されたフラッシュメモリのような不揮発メモリや電池でバックアップされたRAMに同じ機能をもたせてもよい。

【0023】図4は本発明の別の動作波形を示した図で

ある。

【0024】本発明では、ジョブ単位で電源と周波数の制御を行っている。ここで、マイクロプロセッサがOSの実行も行う場合には、ジョブの内容には、アプリケーションを実行するタスクとその前後に実行されるOSの実行が含まれる。すなわち、図4にジョブ1に関して図示するように、ジョブ1では、プロセッサはまずOSを実行し、次にOSの命令によりアプリケーションのタスクが実行される。アプリケーションのタスクが終了すると、制御がOSに戻されるので、OSが実行される。この後、実行すべきジョブがない場合にはOSの働きにより、プロセッサはアイドル状態になる。アイドル状態とはプロセッサのポインタが特定のメモリの番地をアクセスしつづけている状態である。ここから復帰するには割り込みが必要である。

【0025】図5は、図4に示した動作波形をOSの働きなしに実現するシステムのハードウェア構成を示した図である。

【0026】図5のハードウェア構成では、図3のハードウェア構成に加えて半導体集積回路装置の中にアイドルレジスタ9と比較器10が新たに挿入されている。

【0027】本実施例の動作は以下のようなものである。システム立ち上げ時にブートROM3からアイドルレジスタ9とプロセッサ8とにアイドルアドレスが読み込まれる。また、 $t_a$ レジスタ5には時間 $t_a$ の値が読み込まれる。プロセッサ8は、時刻 $t_0$ にまず、ジョブ1の実行を開始する。このとき、プロセッサ8は、低い動作周波数 $f_L$ と低い電源電圧 $V_L$ でジョブ1を実行する。ジョブ1にはOSの実行、タスク1の実行、さらにOSの実行が含まれる。ただし、ハードウェアはそれらを区別することはできない。ジョブ1の開始後、制御回路4はタイマー6によりジョブの経過時間を測定しはじめる。一定の時間 $t_a$ が経過した時刻 $t_1$ で制御回路4は電源電圧 $V_{dd}$ とクロック周波数 $f$ とを変化させ、プロセッサ8は高い動作周波数 $f_H$ と高い電源電圧 $V_H$ でジョブ1を実行する。最後に時刻 $t_2$ でジョブ1の実行が終了すると、OSはアイドル状態を指定してアイドル状態に入る。このとき、OSはあらかじめブートROM3から入力されたアイドルアドレスにポインタをセットすることによって、アイドル状態になる。したがって、制御回路4はアイドルレジスタに格納されたアドレスとポインタが指定するアドレスとが一致したことを判定することにより、アイドル状態に入ったことを認識する。一致判定を受けて、制御回路4は周波数 $f$ と電源電圧 $V_{dd}$ を周波数 $f_L$ と電源電圧 $V_L$ にそれぞれ落とし、半導体集積回路装置は電力削減モードに入る。次のジョブ（ジョブ2）が開始される場合には、ポインタがアイドル状態とは異なるアドレスを指すのでこれを比較回路10で検知し、制御回路4はジョブの実行時間を測定し始める。ジョブ2終了、ジョブ3の開始と終了も同様な方法で判定することが可能であ

る。

【0028】このように構成することで、OSに依存せずにアイドル状態を認識して、本発明の低電力動作を実行することが可能になる。

【0029】図6は本発明のさらに別の動作波形を示した図である。この例はジョブが終了した後にスリープ状態に入る場合を示したものである。スリープ状態はプロセッサ8がPLLを停止して割り込み待ち状態に入ることである。スリープ状態はクロックが止まっているのでその間に消費電力を消費せず、図の電力のグラフで示すように、アイドル状態を利用するよりも全体の電力を低減することが可能になる。

【0030】図7は、図6に示した動作波形をOSの働きなしに実現するシステムのハードウェア構成を示した図である。

【0031】図7のハードウェア構成は、図5のハードウェア構成とほぼ同様であるが、プロセッサ8が発生するPLL7のon/off信号を制御回路にも入力される。この信号により、スリープ状態であるかどうかを認識することが可能である。したがって、制御回路4はPLL7がオンになったときにジョブが開始されたと認識し、一定の時間 $t_a$ が経過した時点で、電源電圧と周波数を増加させるように電源回路2とPLL7を制御することができる。なお、スリープ状態において、低消費電力化の効果は少なくなるが通常状態への復帰を速くできるように、PLL7を停止させることなく待機時にクロックの供給のみを止めるようにしてもよい。その制御信号を用いて同様の構成とすることが可能であり、これは以下に示す例でも同様である。

【0032】図8は、半導体集積回路装置の各回路の動作の性質を鑑みて、本発明の波形をより詳細に示した図である。図8に示すように、1つのジョブはPLLのセトリング時間、OSの実行、低周波数 $f_L$ でのタスクの実行、低周波数 $f_L$ でのタスク実行中の電源電圧の変更、高周波数 $f_H$ でのタスクの実行、OSの実行からなっている。

【0033】プロセッサがスリープの時には電源電圧は $V_L$ でかつPLLは停止している。割り込みにより、この状態から抜け出そうとするときには、まずPLLを動作させようとする。図7のハードウェアではPLLの制御信号を制御回路が受け取るので、割り込みが発生した時からカウンタによりジョブの実行時間を測定し始める。しかしPLLは通常いきなり動作が安定するわけではなく、通常数10マイクロ秒ほどセトリング時間を必要とする。セトリング時間の間に周波数は $f_L$ にセットされる。セトリング時間経過後、まずOSが $f_L$ で実行される。その後、OSによって発行されたタスクが実行される。当初タスクは周波数 $f_L$ で実行されるが、一定の時間 $t_a$ が経過してもジョブが続いている場合には、制御回路4からの信号により、まず電源電圧を $V_L$ から

らVHに移行させる。電源電圧がVHに移行して安定したら次に動作周波数をfLからfHに移行させる。このようにすれば、プロセッサが、低い電源電圧VLを供給される一方で高い周波数fHで駆動されることによって生じる誤動作を防止することが可能になる。タスクが終了すると、OSが周波数fHのまま動作し、OSの指令によりスリープ状態に移行する。

【0034】このように電源電圧と周波数とを制御することにより、半導体集積回路装置内の各回路の特性に応じて、本発明を誤動作なしに実行することが可能になる。

【0035】図9は本発明のさらに別の動作波形を示した図である。周波数がfL, fM, fHの3種類、電源電圧がそれぞれの周波数に対応してVL, VM, VHの3種類になったときの波形図である。このように、本発明の方式は、電源と周波数がそれぞれ2種類には限定されない。この場合、あらかじめ周波数fL、電源電圧VLで動作する期間であるtaと、周波数fL及びfM、電源電圧VL及びVMで動作する期間であるtbの2種類の時間を設定しておく。本波形図に示すように、ジョブ1は時間t0において周波数fLと電源電圧VLで開始される。その後時間taを経過してもジョブ1が終了しないので、周波数fMと電源電圧VMに変えて動作を続け、その周波数と電源電圧で動作を終了する。次のジョブ2は時間t3において、ジョブ1と同様、周波数fLと電源電圧VLで開始する。その後時間taを経過してもジョブが終了しないので、周波数fMと電源電圧VMに変えて動作を続ける。あらかじめ決めておいた時間tbを経過しても終了しないので、周波数fHと電源電圧VHで実行を続け、ジョブ2を終了する。このように、電源電圧と周波数の種類を増やし、きめ細かな制御が可能である。また、電源電圧を2種類、周波数を3種類、というように電源電圧と周波数が1対1に対応している必要もない。ただし、図8に関連して述べたように、供給されている電源電圧で可能な動作速度以上に高速な周波数を供給することのないようにモード（電源電圧と周波数との組み合わせ）を設定する必要がある。

【0036】図10は本発明を特定のアプリケーションの実行に用いた場合の動作波形を示した図であり、本発明の効果を具体的に説明するための実施例である。

【0037】近年、携帯電話等の携帯機器においては、動画像のデコード、エンコードが行われるようになってきた。動画像のデコード、エンコードをマイクロプロセッサで行う場合の特徴としては、そのジョブが周期的に行われることと、ほとんどのジョブは軽いジョブなのに対し、たまに非常に大きな計算量を必要とするジョブが存在することである。

【0038】たとえば、30フレーム/秒のMPEGデータのデコードの場合、そのジョブの実行の周期は33msになる。MPEGのデータ圧縮方式において、第一フレームは画面全体が圧縮された情報になっているので

これをデコードするにはかなりの計算量を要する。一方、第二フレーム以降、しばらくの間はフレーム画像間に連続性が存在し、前のフレームとの差分だけを計算すればいいので、計算量は比較的少ない。本実施例ではこの様子を図示している。

【0039】第一フレームをジョブ1で処理する。最初、周波数fL、電源電圧VLで処理が開始されるが、ジョブ1は計算量が多いので、時間taが経過すると周波数fH、電源電圧VHに変更して処理を継続し、処理を終了する。ここで、周波数fLは周波数fHの1/2とし、時間taを周期T(33ms)の1/2である17.5msとする。時間taまでの周波数はfH/2、時間ta～時間Tの周波数はfHとなるので、周期Tでの平均の周波数は3fH/4となる。したがって、本方式によれば、周波数fHがマイクロプロセッサの達成しうる最大の周波数と仮定すると、平均してその75%の周波数で動作したのと同じになる。

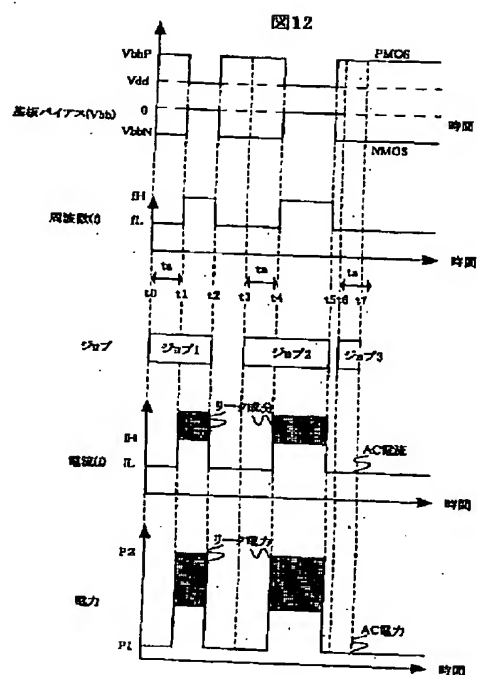
【0040】一方、第二フレーム、第三フレーム、第四フレームの計算をそれぞれジョブ2、ジョブ3、ジョブ4とすると、それぞれの計算量は少ないので、それぞれ時間ta以内に計算を終えることができる。したがって、第二フレーム以降の計算は周波数fLと電源電圧VLを維持できる。この間の電力は最大電力の1/8(12.5%)に押さえることができる。このように、本発明によれば低電力モード時の電力は、最大時の12.5%に押さえながら、マイクロプロセッサが最大に発揮しうる性能の75%を達成することが可能である。

【0041】図11はマルチタスクOSを使って複数のアプリケーションが同時にCPU上を走る場合の動作波形を示したものである。

【0042】マルチタスクOSの場合は、アプリケーションはそれぞれジョブに分割されて、プロセッサにより実行される。各ジョブの計算量はアプリケーションの性質によるところが大きい。たとえば、アプリケーションがそれぞれ圧縮された動画像のデコード、圧縮された音声のデコードであった場合、動画像のデコードの方がより大きな計算量を必要とするので分解されたジョブの長さが長くなることが予想される。一方で、動画像のデコードはジョブ毎のばらつきの幅が大きい。このような場合、動画像のアプリケーションには周波数と電源電圧を変えるまでの設定時間を比較的長くしておけば、設定時間に達する前にたいいていのジョブは終了して低電力化に貢献できる。

【0043】図11の例では、ジョブ1とジョブ3はアプリケーションAに対するジョブであり、ジョブ2とジョブ4はアプリケーションBに対するジョブである。この時、アプリケーションAのジョブ1に対してはより短い設定時間taを適用し、アプリケーションBのジョブに対しては、より長い設定時間tbを適用する。このように、マルチタスクの場合にはアプリケーションの性質に

【図12】



フロントページの続き

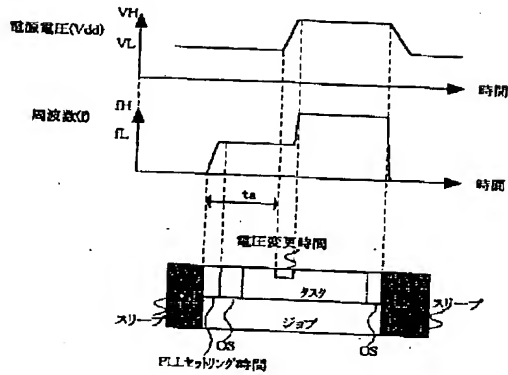
F ターム(参考) 5B011 DC06 EA08 LL02 LL13  
 5B062 AA03 AA05 GG01 HH02 HH06  
 5B079 AA07 BA01 BB01 DD13

30

BEST AVAILABLE COPY

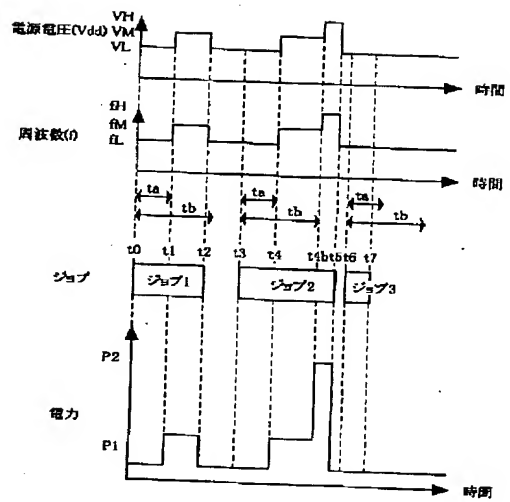
【図8】

図8



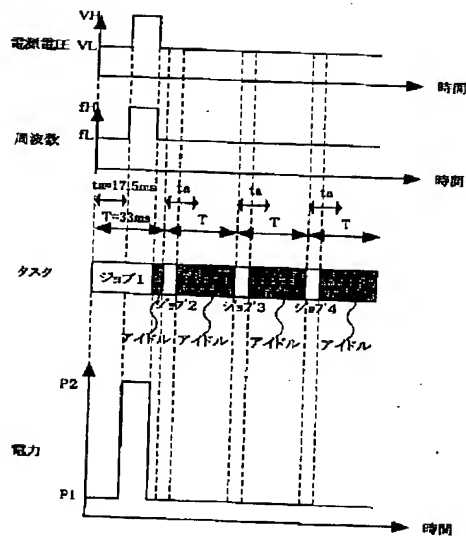
【図9】

図9



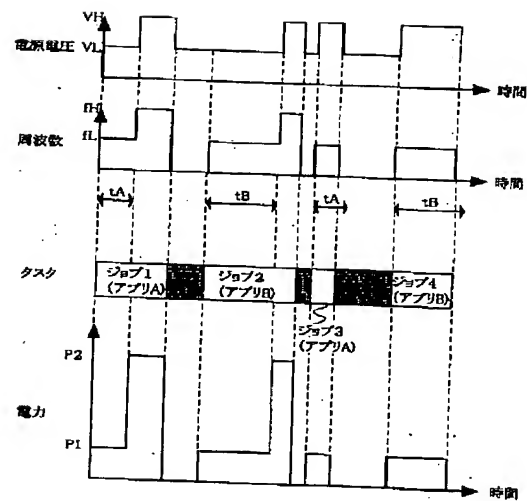
【図10】

図10



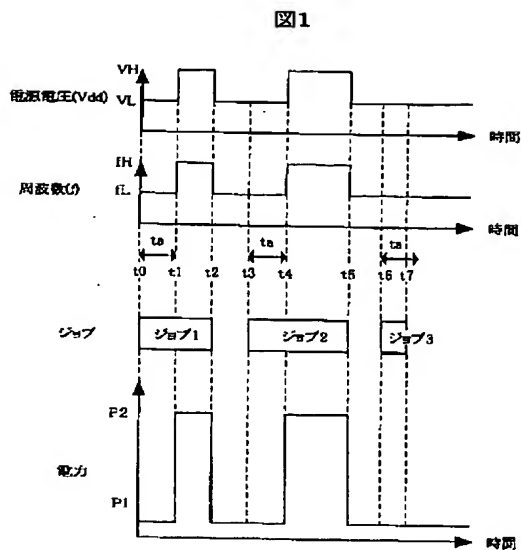
【図11】

図11

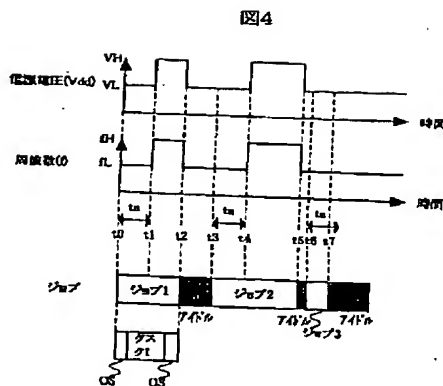




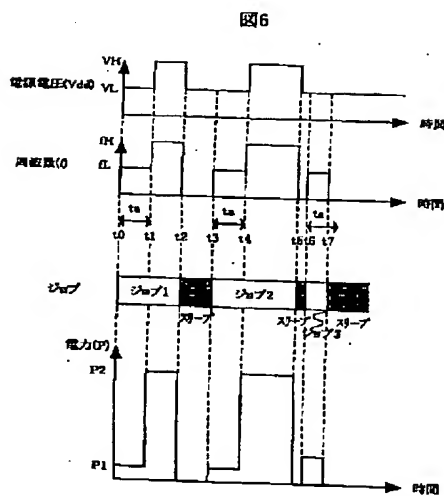
【図1】



【図4】

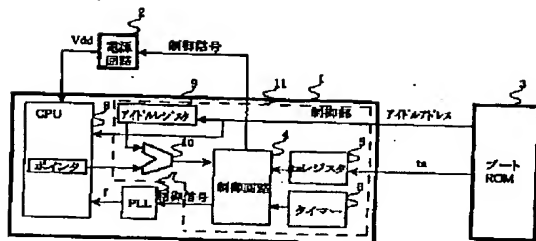


【図6】



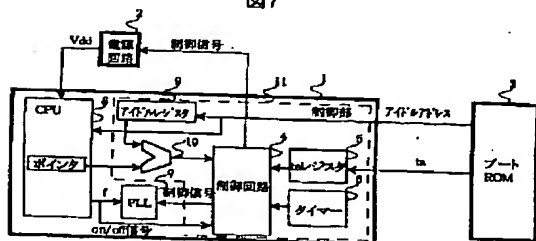
【図5】

図5



【図7】

図7



BEST AVAILABLE COPY

応じた設計時間を設定することにより、効果的に低消費電力化することができる。この場合には、 $t_{\text{a}}$ レジスタ

(図2、図3、図5、図7)には設定時間 $t_{\text{a}}$ の情報と設定時間 $t_{\text{b}}$ の情報がそれぞれ必要になる。

【0044】図12は本発明の第2の実施例として電源電圧を一定として、周波数と同時に基板バイアスを制御した場合の波形図である。

【0045】図12に示す例では、ジョブの開始時にはプロセッサに対して低い周波数 $f_{\text{L}}$ と深い基板バイアスを供給する。その後、時間 $t_{\text{a}}$ が経過した後は、高い周波数 $f_{\text{H}}$ と浅い基板バイアスを供給する。深い基板バイアスとしては、プロセッサを構成するNMOSトランジスタの基板またはウェルに $V_{\text{bbNV}}$ 、PMOSトランジスタの基板またはウェルに $V_{\text{bbPV}}$ を印加する。浅い基板バイアスとしては、深い基板バイアスとして印加した基板電位の絶対値よりも小さい絶対値の基板電位を印加する。例えば、NMOSトランジスタの基板またはウェルには0V、PMOSトランジスタの基板またはウェルに電源電位( $V_{\text{dd}}$ )を印加する。

【0046】深い基板バイアスを印加した場合には、しきい値電圧が上昇して動作周波数が下がるがリーク電流が減るので電力が低減される。一方、浅い基板バイアスを印加した場合はしきい値電圧が下がって動作周波数が上昇するがリーク電流が増えて電力が増加する。この動作速度と消費電力との関係は周波数と電源電圧のとを制御した場合と同様である。したがって、図12の制御例によってもOSに依存せずに汎用性の高い周波数制御方式を実現し、マイクロプロセッサの低電力化と高性能化を両立させることが可能になる。

【0047】なお、この第2の実施例についても図2、図3、図5、図7に示したようなハードウェア構成により実現可能である。すなわち、電源回路に代えて基板バイアス発生回路を設け、図12に示すような基板バイアスを発生させるようにすればよい。

【0048】さらに、第1の実施例として電源電圧と周波数を制御する例を示したが、周波数のみの制御も可能である。低消費電力の効果は第1の実施例に比べて小さ

くなるが、簡易に構成ができる。この場合には、図2、図3、図5、図7に示したようなハードウェア構成から電源回路への制御を除いた形で実現することができる。

【0049】

05 【発明の効果】本発明によれば、OSに依存せずに汎用性の高い周波数制御方式を実現し、マイクロプロセッサの低電力化と高性能化を両立させることが可能になる。

【図面の簡単な説明】

10 【図1】本発明の第1の実施例である電源-周波数制御の例を示す波形図である。

【図2】本発明を実施するためのハードウェアの構成例を示す図である。

【図3】本発明を実施するためのハードウェアの別の構成例を示す図である。

15 【図4】本発明の第1の実施例である電源-周波数制御の別の例を示す波形図である。

【図5】本発明を実施するためのハードウェアの別の構成例を示す図である。

20 【図6】本発明の第1の実施例である電源-周波数制御の別の例を示す波形図である。

【図7】本発明を実施するためのハードウェアの別の構成例を示す図である。

【図8】本発明の第1の実施例である電源-周波数制御の波形図を詳細に示した図である。

25 【図9】本発明の第1の実施例である電源-周波数制御の別の例を示す波形図である。

【図10】本発明を特定のアプリケーションの実行に用いた場合の波形図である。

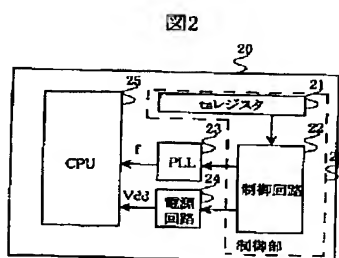
30 【図11】マルチタスクOSにより複数のアプリケーションが同時にCPU上を走る場合の波形図である。

【図12】本発明の第2の実施例である基板バイアス-周波数制御の例を示す波形図である。

【符号の説明】

1：半導体集積回路装置、2：電源回路、3：ブートROM、4：制御回路、5：レジスタ、6：タイマー、7：クロック供給回路、8：プロセッサ、9：アイドルレジスタ、10：比較器、11：制御部。

【図2】



【図3】

